

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平8-185164

(43) 公開日 平成8年(1996)7月16日

(51) Int.Cl.⁶

G10H 1/00

識別記号

102 Z

庁内整理番号

F I

技術表示箇所

審査請求 未請求 請求項の数4 F D (全20頁)

(21) 出願番号 特願平6-340418

(22) 出願日 平成6年(1994)12月29日

(71) 出願人 000001443

カシオ計算機株式会社

東京都新宿区西新宿2丁目6番1号

(72) 発明者 瀬戸口 克

東京都羽村市栄町3丁目2番1号 カシオ

計算機株式会社羽村技術センター内

(74) 代理人 弁理士 大菅 義之

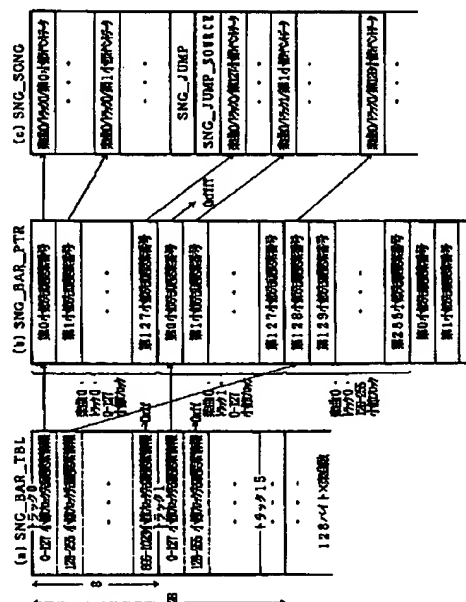
(54) 【発明の名称】 自動演奏制御装置及びそれに使用する楽曲データ記憶装置

(57) 【要約】

【目的】 シーケンサー等のメモリに格納された楽曲データを自動演奏するための自動演奏制御技術に関し、編集作業を簡単に実行し、小節単位のジャンプ操作なども迅速に実行可能とすることを目的とする。

【構成】 配列 SNG_BAR_TBL には、楽曲毎及び各楽曲内のトラック毎に、第0小節～第1023小節の1024小節を128小節ずつ分割して得られる8つの小節ブロックのそれぞれに対応する、配列 SNG_BAR_PTR 内の128個ずつの配列データ群の各先頭要素情報が記憶される。配列 SNG_BAR_PTR には、小節ブロック毎に、各小節ブロック内の128小節のそれぞれに対応する、配列 SNG_SONG 内のイベントデータ群の各先頭要素番号が記憶される。例えば小節単位の早送り再生又は巻き戻し再生時には、配列 SNG_BAR_TBL 上の小節ブロック又は配列 SNG_BAR_PTR 上の小節に対する各ポインタ値をインクリメント又はデクリメントするだけでよい。

RAM105に記憶される楽曲データの
データフォーマットを示す図(その2)



【特許請求の範囲】

【請求項1】 自動演奏を行うための楽曲データに対して自動演奏のための制御を行う自動演奏制御装置において、

小節単位の楽曲データ群であって、それぞれ小節の先頭からの時間データを含むことのできる楽曲データ群を記憶する第1の記憶手段と、

前記小節単位の楽曲データ群の前記第1の記憶手段上での記憶位置を示す情報である小節単位楽曲データ群指示データを記憶する第2の記憶手段と、

該第2の記憶手段に記憶される小節単位楽曲データ群指示データを操作することによって、前記第1の記憶手段に記憶される楽曲データ群に対して自動演奏のための制御を実行する制御手段と、

を有することを特徴とする自動演奏制御装置。

【請求項2】 前記第2の記憶手段に記憶される小節単位楽曲データ群指示データは、連続して記憶される複数個ずつが小節ブロックとして管理され、

該小節ブロック単位の小節単位楽曲データ群指示データの前記第2の記憶手段上での記憶位置を示す情報である指示データを記憶する第3の記憶手段を更に有し、

前記制御手段は、該第3の記憶手段に記憶される指示データ及び前記第2の記憶手段に記憶される小節単位楽曲データ群指示データを操作することによって、前記第1の記憶手段に記憶される楽曲データ群に対して自動演奏のための制御を実行する、

ことを特徴とする請求項1に記載の自動演奏制御装置。

【請求項3】 自動演奏を行うための楽曲データに対して自動演奏のための制御を行う自動演奏制御装置に使用する楽曲データ記憶装置において、

小節単位の楽曲データ群であって、それぞれ小節の先頭からの時間データを含むことのできる楽曲データ群を記憶する第1の記憶手段と、

前記小節単位の楽曲データ群の前記第1の記憶手段上での記憶位置を示す情報である小節単位楽曲データ群指示データを記憶する第2の記憶手段と、

を有することを特徴とする楽曲データ記憶装置。

【請求項4】 前記第2の記憶手段に記憶される小節単位楽曲データ群指示データは、連続して記憶される複数個ずつが小節ブロックとして管理され、

該小節ブロック単位の小節単位楽曲データ群指示データの前記第2の記憶手段上での記憶位置を示す情報である指示データを記憶する第3の記憶手段を更に有する、

ことを特徴とする請求項3に記載の楽曲データ記憶装置。

【発明の詳細な説明】

【0001】

【産業上の利用分野】 本発明は、シーケンサー等のメモリに格納された楽曲データを自動演奏するための自動演奏制御技術に関する。

【0002】

【従来の技術】 従来、楽曲を電子楽器などに自動演奏させるための自動演奏装置においては、楽曲データが以下のような一般的なデータフォーマットでメモリに記憶されている。即ち、例えばスタンダードMIDIファイルでは、ノートオン、ノートオフなどのイベント間の時間（デルタタイム）が、イベントデータの間に挟まれたデータフォーマットを有する。

【0003】 また、ノートオンからノートオフまでの時間はゲートタイムとして有し、次のノートオン又は他のイベントまでの時間をステップタイムの形式で有するデータフォーマットも知られている。

【0004】 しかし、これらのデータフォーマットは、いずれもイベントデータが単に再生されるのみであれば手軽でよい。

【0005】

【発明が解決しようとする課題】 しかし、イベントデータに対して編集操作を実行したり、ランダムな小節へのジャンプ操作を実行しようとした場合に、上述のデータフォーマットに対してこれらの操作に対応するデータ処理を実行するには、非常に煩雑かつ複雑な制御を要するという問題点を有している。

【0006】 本発明の課題は、編集作業を簡単に実行し、小節単位のジャンプ操作なども迅速に実行可能とすることにある。

【0007】

【課題を解決するための手段】 本発明は、自動演奏を行うための楽曲データに対して、楽曲データの記録、再生、編集などの自動演奏のための制御を行う自動演奏制御装置を前提とする。

【0008】 そして、まず、小節単位の楽曲データ群であって、それぞれ小節の先頭からの時間データを含むことのできる楽曲データ群（イベントデータ群）を記憶する第1の記憶手段（配列SNG_SONGを記憶するRAM105）を有する。

【0009】 次に、小節単位の楽曲データ群の第1の記憶手段上での記憶位置を示す情報である小節単位楽曲データ群指示データを記憶する第2の記憶手段（配列SNG_BAR_PTRを記憶するRAM105）を有する。

【0010】 そして、その第2の記憶手段に記憶される小節単位楽曲データ群指示データを操作することによって、第1の記憶手段に記憶される楽曲データ群に対して自動演奏のための制御を実行する制御手段（CPU103等）を有する。

【0011】 本発明は、上述の発明の構成に加えて、更に次のような構成を有することができる。まず、第2の記憶手段に記憶される小節単位楽曲データ群指示データは、連続して記憶される複数個ずつが小節ブロックとして管理される。

【0012】次に、その小節ブロック単位の小節単位楽曲データ群指示データの第2の記憶手段上での記憶位置を示す情報である指示データを記憶する第3の記憶手段（配列SNG_BAR_TBLを記憶するRAM105）を更に有する。

【0013】そして、前述の制御手段は、その第3の記憶手段に記憶される指示データ及び第2の記憶手段に記憶される小節単位楽曲データ群指示データを操作することによって、第1の記憶手段に記憶される楽曲データ群に対して自動演奏のための制御を実行する。

【0014】なお、上記第1、第2、又は第3の記憶手段のみからなる楽曲データ記憶装置も、本発明の範囲である。

【0015】

【作用】例えば楽曲データの再生処理において、小節単位の早送り再生又は巻き戻し再生が実行される場合には、制御手段が、第2の記憶手段に記憶される小節単位楽曲データ群指示データ又は第3の記憶手段に記憶される指示データをインクリメント又はデクリメントするだけでよい。この結果、非常に高速な1又は複数小節単位の早送り再生又は巻き戻し再生を簡単な制御で実現することができる。

【0016】次に、楽曲データの編集処理が実行される場合を考える。例えば小節単位の移動を行うためには、制御手段が、第2の記憶手段に記憶される小節単位楽曲データ群指示データ又は第3の記憶手段に記憶される指示データの並びを入れ替えるだけでよい。この場合、各小節の楽曲データ群に記憶される時間データはそれぞれの小節の先頭からの時間データとすることができるため、各楽曲データの時間データを編集し直す必要もない。

【0017】更に、例えば既存の小節に新たな楽曲データを追加するためには、制御手段は、第1の記憶手段の末尾に楽曲データ群を追加し、それに対応するデータを第2又は第3の記憶手段に設定すればよい。

【0018】その他、楽曲データの挿入、削除等も、簡単なデータ操作で実現できる。このように、楽曲データが、第1の記憶手段、第2の記憶手段、又は第3の記憶手段を介して階層的に管理されることにより、楽曲データの再生処理、編集処理等の制御を簡単に実行することができる。

【0019】

【実施例】以下、図面を参照しながら本発明の実施例につき詳細に説明する。図1は、本発明の実施例の全体構成図である。この実施例は、シーケンサ装置として実現されている。

【0020】CPU103は、ROM104に記憶されている制御プログラムに基づいて、システム全体の動作を制御する。そして、楽曲データの記録時には、特に図示しない電子楽器から出力された演奏情報が、例えば

MIDI(Musical Instrument Digital Interface)規格に基づいて転送され、その演奏情報が演奏情報入力部101によって受信される。CPU103は、この演奏情報を取り込み、その演奏情報を示すイベントデータを、必要に応じてその入力タイミングと共に、RAM105の楽曲データ記録領域上の記録対象トラックに順次記録する。記録対象トラックは、ユーザが予めシステム操作子群107を操作して指定する。

【0021】楽曲データの編集時には、CPU103は、ユーザがシステム操作子群107において指定した内容に従って、RAM105の楽曲データ記録領域上の所望のトラックに記録されている楽曲データを読み出して、例えば5線譜の形式で表示部102に表示する。ユーザは、システム操作子群107を用いて、表示部102に表示されている楽曲データに対して、楽曲データの挿入、削除、移動、複写などを行うことができる。この編集結果は、RAM105の楽曲データ記録領域の楽曲データの内容に反映される。

【0022】楽曲データの再生時には、RAM105の楽曲データ記録領域上の再生対象トラックから、イベントデータが順次読み出され、各イベントデータに対応する演奏情報が、例えばMIDI規格に基づいて演奏情報出力部106から外部の電子楽器又は音源装置に出力され、これらの装置によってその演奏情報に対応する自動演奏が行われる。

【0023】図2～図5は、RAM105に記憶される楽曲データのデータフォーマットを示す図である。本実施例では、楽曲データは、各小節のイベントデータと、連続する128小節分のイベントデータを単位とする小節ブロックと、最大で8個の小節ブロックを含むトラック、16トラックを単位とする楽曲という各单位を用いて、階層的に管理される点が特徴である。

【0024】まず、図2に示されるトラック使用状況データSNG_TRK_USEは、16ビットのデータであって、各楽曲毎に設定される。各データには、トラック0～15の各トラックが楽曲の自動演奏において使用されるか否かが、“1”又は“0”のフラグによって設定される。このデータの、最下位ビットがトラック0に対応し、最上位ビットがトラック15に対応する。

【0025】図3(a)に示される配列SNG_BAR_TBLには、楽曲毎及び各楽曲内のトラック毎に、第0小節～第1023小節の1024小節を128小節ずつ分割して得られる8つの小節ブロックのそれぞれに対応する、図3(b)に示される配列SNG_BAR_PTR内の128個ずつの配列データ群の各先頭要素情報が記憶される。配列SNG_BAR_PTRに定義可能な小節ブロックの数は、例えば最大で32である。その場合、配列SNG_BAR_TBLには、小節ブロック先頭要素情報として、0x00～0x1f（10進数の0～31に対応）の何れかの値が記憶される。この小節ブ

ロック先頭要素情報の値を128倍して得られる値が、その小節ブロックに対応する配列SNG_BAR_PTR内の128個の配列データ群の先頭要素番号となる。但し、未使用のトラックのそれぞれの小節ブロックの先頭要素情報及び使用トラック内の未使用の小節ブロックの先頭要素情報としては、値0x f fが設定される。

【0026】図3(b)に示される配列SNG_BAR_PTRには、上述の小節ブロック毎に、各小節ブロック内の128小節のそれぞれに対応する、図3(c)に示される配列SNG_SONG内のイベントデータ群の各先頭要素番号が記憶される。配列SNG_SONGに定義可能なイベントデータの数、例えば最大で16384イベントである。その場合、配列SNG_BAR_PTRには、各小節に対応するイベントデータ群の先頭要素番号として、0x0000~0x4000(10進数の0~16383に対応)の何れかの値が記憶される。但し、全休符の小節に対応する先頭要素番号としては、値0x f f f fが設定される。

【0027】図3(c)に示される配列SNG_SONGには、上述したように小節単位で連続したイベントデータが記憶される。1つのイベントデータは、後述するように3ワードのデータとして構成される。但し、配列SNG_SONGは、512個のイベントデータを単位とする32個のイベントデータブロック(16384イベントデータ分)に分割される。ここで、1つの小節内のイベントデータ群が2つのイベントデータブロックに分割される場合には、第1番目のイベントデータブロックの末尾に後述するジャンプイベントデータが記憶され、また、第2番目のイベントデータブロックの先頭に後述するジャンプソースイベントデータが記憶される。

【0028】図4は、図3(c)の配列SNG_SONGに記憶されるイベントデータ群を構成するノートイベントデータとコントロールイベントデータのデータフォーマットを示す図であり、各イベントデータは3ワードのデータとして構成される。

【0029】図4(a)に示されるノートイベントデータは、楽譜の音符に対応するものであり、単音の発音処理に必要な複数の情報を含む。即ち、ノートイベントデータの第1ワードは、上位7ビットが音高番号、下位7ビットがベロシティを表わす。ノートイベントデータの第2ワードは、そのノートイベントデータに対応する音符について、それが含まれる小節の先頭からの時間位置を、適当な単位(例えば480分の1拍のクロック)でカウントした値である。そして、ノートイベントデータの第3ワードは、そのノートイベントデータに対応する音符の発音時間を前述の適当な単位でカウントしたゲートタイムである。

【0030】図4(b)に示されるコントロールイベントデータは、楽曲の再生を制御するために必要な、上述のノートイベントデータ以外のイベントデータであり、第

1ワードの最上位ビットが1であることによって、ノートイベントデータと区別される。このコントロールイベントデータは、更に、メタイイベントデータ群、ジャンプイベントデータ群、及びコマンドイベントデータ群の3つに分類される。メタイイベントデータ群は、楽曲全体のテンポ、拍子、調に関する情報を含む。ジャンプイベントデータ群は、図5で後述するように、トラック終了、小節終了、次処理データへのジャンプ等の、イベントデータの位置制御に関する情報を含む。更に、コマンドイベントデータ群は、音量、プログラムチェンジ(音色)、ベンダー等の、楽音に対する修飾的な効果を付加するための情報を含む。コントロールイベントデータの第1ワードの下位7ビットはコントロールコードを示し、このコードによって制御内容が決定される。例えば、メタイイベントデータ群のテンポイベントデータに対応する第1ワードの値は0x f f 0 0、ジャンプイベントデータ群の小節終了イベントデータに対応する第1ワードの値は0x f f 1 1(図5(b)参照)、コマンドイベントデータ群のプログラムチェンジイベントデータに対応する第1ワードの値は0x f f 2 0となる。コントロールイベントデータの第2ワードは、そのコントロールイベントデータに対応する制御が実行される小節の先頭からの時間位置を、前述した適当な単位でカウントした値である。コントロールイベントデータの第3ワードには、コントロールコード以外に必要なコントロール値が格納される。例えばメタイイベントデータ群のテンポイベントデータの第3ワードにはテンポ番号、ジャンプイベントデータ群のジャンプイベントデータの第3ワードには配列SNG_SONG(図3(c))内でのジャンプ先のイベントデータの要素番号、コマンドイベントデータ群のプログラムチェンジイベントデータの第3ワードには音色番号が格納される。

【0031】次に、上述したコントロールイベントデータのうちのジャンプイベントデータ群を構成する4種類のイベントデータについて説明する。図5は、これら4種類のイベントデータのそれぞれのデータフォーマットを示す図である。

【0032】まず、図5(a)のトラック終了イベントデータは、各トラックのイベントデータ群の終了位置にそのトラックの最終のイベントデータとして挿入される。次に、図5(b)の小節終了イベントデータは、各小節のイベントデータ群の終了位置にその小節の最終イベントデータとして挿入される。

【0033】更に、図5(c)のジャンプイベントデータと図5(d)のジャンプソースイベントデータは、図3(c)の配列SNG_SONGにおいて、1つの小節内のイベントデータ群が512イベントずつの2つのイベントデータブロックに分割される場合に、それぞれ、第1番目のイベントデータブロックの末尾と第2番目のイベントデータブロックの先頭に記憶される。この場合、ジ

ジャンプイベントデータの第3ワードには配列SNG_SONG内でのジャンプ先のイベントデータの要素番号が格納され、ジャンプソースイベントデータの第3ワードには同じくジャンプ元のイベントデータの要素番号が格納される。

【0034】なお、図5(a)～(d)に示される4種類のイベントデータについて、イベントデータの編集処理及び再生処理においては、第1ワードにより現在処理イベントデータがジャンプイベントデータ群に属するイベントであることが認識された時点で、各々のイベントデータに対応した処理が即座に実行される。従って、これら各イベントデータの第2ワードは任意の値でよい。また、トラック終了イベントデータと小節終了イベントデータの第3ワードも任意の値でよい。

【0035】以上説明した図2～図5の楽曲データのデータフォーマットが採用されることにより、自動演奏に関する制御を非常に効率的に実行することが可能となる。この効果について、以下に説明する。

【0036】まず、イベントデータの再生処理時(楽曲の自動演奏時)の動作について説明する。ユーザが図1のシステム操作子群107を操作することにより再生開始が指示されると、図1のCPU103は、RAM105に記憶されている再生される楽曲に対応する図2に示されるトラック使用状況データSNG_TRK_USEの内容を調べる。CPU103は、トラック使用状況データSNG_TRK_USEにおいて値1が設定されているビットに対応するトラックの処理タイミングで、そのトラックのイベントデータを再生させるために、まず、図3(a)に示される配列SNG_BAR_TBLを探索する。この結果、CPU103は、現在再生中の小節に対応する図3(b)に示される配列SNG_BAR_PTR内の要素番号を得る。次に、配列SNG_BAR_PTR内の該当データの値が0xffffでなければ(全休符小節でなければ)、CPU103は、配列SNG_BAR_PTRの該当データの値が指す要素番号の配列SNG_SONG中のイベントデータを読み込む。そして、その第2ワードに格納されている小節先頭からのクロックの値が現在のクロックの値以下であれば、CPU103は、そのイベントデータを再生するために、そのイベントデータの第1ワードのデータに従った演奏情報を、図1の演奏情報出力部106から出力させる。以下、上述の一連の処理が繰り返し実行され、現在小節の現在クロックに従ったイベントデータの再生処理が実行される。

【0037】上述のイベントデータの再生処理において、小節単位の早送り再生又は巻き戻し再生が実行される場合を考える。この動作を実現するためには、図3(a)に示される配列SNG_BAR_TBL上の小節ブロックに対する現在処理ポインタ値、又は図3(b)に示される配列SNG_BAR_PTR上の小節に対する現

在処理ポインタ値をインクリメント又はデクリメントするだけでよい。この結果、非常に高速な1又は複数小節単位の早送り再生又は巻き戻し再生を簡単な制御で実現することができる。

【0038】次に、イベントデータの編集処理が実行される場合を考える。例えば小節単位の移動を行うためには、図3(a)の配列SNG_BAR_TBL又は図3(b)の配列SNG_BAR_PTRの配列データの要素を入れ替えるのみでよい。この場合、図3(c)の配列SNG_SONG内のイベントデータに記憶されているクロックデータは小節の先頭からのクロックであるため(図4等参照)、時間データを編集し直す必要もない。更に、例えば既存の小節に新たなイベントデータを追加するためには、図3(c)の配列SNG_SONGの末尾にイベントデータ群を追加し、ジャンプイベントデータとジャンプソースイベントデータ(図5参照)を用いて、そのイベントデータ群を所望の小節のイベントデータ群に連結するだけでよい。その他、イベントデータの挿入、削除等も、簡単なデータ操作で実現できる。

【0039】このように、楽曲データが階層的に管理されることにより、イベントデータの再生処理、編集処理等の制御を簡単に実行することができる。ここで、上述の楽曲データの階層構造が採用される場合におけるイベントデータの編集処理において、小節内でイベントデータの挿入、削除、移動等の編集作業が実行された場合、図3(c)の配列SNG_SONG内の1つの小節内のイベントデータ群において、図5に示されるジャンプイベントデータ群以外のイベントデータの第2ワードに設定されている小節の先頭からのクロックの値の大小関係が逆転している状況が発生することがある。例えば、所望の小節内の任意の時間位置に新たなイベントデータが追加される場合、配列SNG_SONGの末尾に、ジャンプソースイベントデータと、それに続いて、第2ワードに所望のクロック値が設定された所望のイベントデータと、更にそれに続いて、小節終了イベントデータが追加されると共に、上記所望の小節の最終イベントの小節終了イベントデータが、第3ワードに上記ジャンプソースイベントデータの要素番号が設定されたジャンプイベントデータに置き換えられ、最後に、上記ジャンプソースイベントデータの第3ワードに上記ジャンプイベントデータの要素番号が設定される。この場合、上記追加イベントデータは、上記所望の小節内のイベントデータ群の最後に単純に追加されただけであるため、その追加イベントデータが上記所望の小節内の他のイベントデータ群との間で正しい時間関係を有するように、その追加イベントデータを含む上記所望の小節内のイベントデータ群が、各イベントデータの第2ワードのクロック値に関して並び換えられる(ソートする)必要がある。

【0040】この場合に、小節内のイベントデータ群に含まれ得る図5に示されるジャンプイベントデータ群が

適切に処理される必要がある。そこで、本実施例においては、小節内のイベントデータ群の時間ソート処理が次のようにして実行される。以下に、小節内のイベントデータ群の時間ソート処理の詳細を説明する。

【0041】図6は、図1に示されるCPU103がROM104に記憶された制御プログラムを実行する動作として実現される時間ソート処理の全体動作フローチャートである。

【0042】まず、時間ソート処理終了フラグ（CPU103内のレジスタ又はRAM105に確保される）は、その値が0であるときに時間ソート処理が終了していないことを示し、その値が1であるときに時間ソート処理が終了したことを示す。

【0043】まず、ステップ601において、時間ソート処理終了フラグの値が0にセットされる。次に、ステップ602で時間ソート処理終了フラグの値が0であると判定される間、ステップ603～605の一連の処理が繰り返し実行される。

【0044】ステップ603では、時間ソート処理終了フラグの値が1にセットされる。ステップ604では、配列SNG__SONGに記憶されている時間ソート処理の対象とされる小節内のイベントデータ群に含まれ、ジャンプイベントデータ及びジャンプソースイベントデータによって区切られる全てのイベントデータのグループ（以下、イベントデータフラグメント又は単にフラグメントと呼ぶ）のそれぞれの内部で閉じた時間ソート処理であるフラグメント内ソート処理が実行される。

【0045】続いて、ステップ605では、配列SNG__SONGに記憶されている上記対象小節に含まれる隣接するフラグメント同士で、それらの接続部に位置するイベントデータ（ジャンプイベントデータの直前に位置するイベントデータとそのジャンプイベントデータに対応するジャンプソースイベントデータの直後に位置するイベントデータ）についての時間ソート処理であるフラグメント間ソート処理が実行される。

【0046】そして、上述のステップ604のフラグメント内ソート処理とステップ605のフラグメント間ソート処理が繰り返し実行され、両方の処理において実際にイベントデータの入替えが発生しなくなった時点で、配列SNG__SONGに記憶されている対象小節内のイベントデータ群に対する時間ソート処理が完了する。具体的には、ステップ604又は605において実際にイベントデータの入替えが発生した場合は、ステップ604又は605内において後述するように時間ソート処理終了フラグの値が0にセットされる。従って、この場合にはステップ602の判定がYESとなるため、時間ソート処理が続行される。一方、ステップ604及び605の両方で実際にイベントデータの入替えが発生しなくなった場合は、ステップ604及び605の何れにおいても時間ソート処理終了フラグの値を0にセットする処

理が実行されなくなる。従って、この場合には時間ソート処理終了フラグの値はステップ603で1にセットされたままとなるため、ステップ602の判定がNOとなり、時間ソート処理が終了する。

【0047】図7は、図6のステップ604のフラグメント内ソート処理の動作フローチャートである。まず、フラグメント内ソート処理終了フラグ（CPU103内のレジスタ又はRAM105に確保される）は、フラグメント内ソート処理を、終了させるべきでない場合に値0となり、終了させるべきである場合に値1となる。このフラグの値は、ステップ701で、0に初期化される。

【0048】サーチ状態フラグ（CPU103内のレジスタ又はRAM105に確保される）は、配列SNG__SONGに記憶されている対象小節に含まれる現在処理中のフラグメント内において、ジャンプソースイベントデータ以外の先頭のイベントデータをサーチする処理が、終了していない場合に値0となり、終了している場合に値1となる。このフラグの値は、ステップ702で、0に初期化される。

【0049】サーチイベント（CPU103内のレジスタ又はRAM105に確保される）は、配列SNG__SONGに記憶されている対象小節に含まれる現在処理中のフラグメント内において、現在処理中のイベントデータを指示するものである。このサーチイベントの値は、ステップ703で、配列SNG__SONGに記憶される対象小節内の先頭のイベントデータの要素番号にセットされる。この番号は、図3の配列SNG__BAR__TBL及び配列SNG__BAR__PTRから容易に検索できる。

【0050】続いて、初期状態では、ステップ704と705の判定が共にYESとなった後、ステップ706～713において、配列SNG__SONGに記憶されている対象小節に含まれる現在処理中のフラグメント内において、ジャンプソースイベントデータ以外の先頭のイベントデータをサーチする処理が実行される。

【0051】まず、サーチイベントによって指示される現在イベントデータ（フロー中では現在イベントと記述する）が、RAM105内の配列SNG__SONGから読み出される。

【0052】そして、現在イベントデータがジャンプソースイベントデータである場合は、ステップ706の判定の後にステップ707が実行され、サーチイベントが指示する配列SNG__SONGの要素番号がインクリメントされる。これらのステップ706とステップ707の処理の繰返しによって、現在処理中のフラグメント内の先頭のジャンプソースイベントデータが無視される。

【0053】現在イベントデータがジャンプソースイベントデータ以外のイベントデータとなった場合は、ステップ708で、そのイベントデータの種類の判定され

る。ステップ708で現在イベントデータがトラック終了イベントデータ又は小節終了イベントデータ(図5参照)であると判定された場合には、ステップ709で、フラグメント内ソート処理終了フラグの値が1にセットされる。即ち、この場合は、現在処理中のフラグメント内にはソート処理を実行すべき有効なイベントデータは存在せず、かつ現在処理中のフラグメント以降に対象小節のイベントデータ群が存在しないため、ステップ709の処理の後にステップ704が実行された時点でその判定がNOとなり、フラグメント内ソート処理を終了する。

【0054】また、ステップ708で現在イベントデータがジャンプイベントデータ(図5(c)参照)であると判定された場合には、ステップ710で、そのジャンプイベントデータの第3ワードに格納されているジャンプ先要素番号が、サーチイベントの値としてセットされる。即ち、この場合には、現在処理中のフラグメント内にはソート処理を実行すべき有効なイベントデータは存在せず、かつ現在処理中のフラグメント以降に対象小節のイベントデータ群が存在する。このため、ステップ710の処理によってサーチイベントに新たなフラグメントの先頭のイベントデータの要素番号が設定された後に、再びステップ704と705の判定が共にYESとなった後、ステップ706~713において、配列SNG_SONGに記憶されている対象小節に含まれる新たなフラグメント内において、ジャンプソースイベントデータ以外の先頭のイベントデータをサーチする処理が再度実行される。

【0055】更に、ステップ708で現在イベントデータがトラック終了イベントデータ、小節終了イベントデータ、及びジャンプイベントデータ以外のイベントデータであると判定された場合には、ステップ711で、サーチイベントの値がフラグメント先頭イベントの値としてセットされる。この値が、現在処理中のフラグメント内で最初に見つかったジャンプソースイベントデータ以外の先頭のイベントデータの要素番号となる。続いて、ステップ712で、サーチイベントが指示する配列SNG_SONGの要素番号がインクリメントされた後、サーチ処理を終了させるべく、ステップ713で、サーチ状態フラグの値が1にセットされる。

【0056】この結果、ステップ713に続いてステップ704の判定がYESとなった後に、ステップ705の判定がNOとなって、ステップ714~716が実行される。ステップ714~716では、配列SNG_SONGに記憶されている対象小節に含まれる現在処理中のフラグメント内で、トラック終了イベントデータ、小節終了イベントデータ、及びジャンプイベントデータ以外の末尾のイベントデータをサーチする処理が実行される。

【0057】まず、サーチイベントによって指示される

現在イベントデータが、RAM105内の配列SNG_SONGから読み出される。そして、現在イベントデータがトラック終了イベントデータ、小節終了イベントデータ、及びジャンプイベントデータ以外のイベントデータである場合には、ステップ714の判定の後にステップ715が実行され、サーチイベントが指示する配列SNG_SONGの要素番号がインクリメントされる。これらのステップ714とステップ707の処理の繰返しによって、現在処理中のフラグメント内のイベントデータが末尾に向かって順次サーチされる。

【0058】現在イベントデータがトラック終了イベントデータ、小節終了イベントデータ、又はジャンプイベントデータの何れかのイベントデータとなった場合は、ステップ716で、サーチイベントが指示する要素番号から-1して得られる値が、フラグメント終了イベントの値としてセットされる。この値が、現在処理中のフラグメント内における、トラック終了イベントデータ、小節終了イベントデータ、及びジャンプイベントデータ以外の末尾のイベントデータである。

【0059】以上の処理により、現在処理中のフラグメント内でソート処理が実行されるべき先頭のイベントデータと末尾のイベントデータが、フラグメント先頭イベント及びフラグメント終了イベントとして算出される。

【0060】続いて、ステップ717では、上記フラグメント先頭イベントとフラグメント終了イベントが指示する要素番号が一致しているか否かが判定される。ステップ717の判定がYESの場合は、現在処理中のフラグメント内でソート処理が実行されるべきイベントデータの数が1であり、実質的にソート処理を実行する必要はないため、ステップ718のソート処理は実行されずに、ステップ719以降が実行される。ステップ719以降の処理については後述する。

【0061】ステップ717の判定がNOの場合は、ステップ718で、配列SNG_SONGに記憶されている対象小節に含まれる現在処理中のフラグメント内で、フラグメント先頭イベントとフラグメント終了イベントにより指示される範囲のイベントデータ群が、各イベントデータの第2ワードのクロック値に関してソートされる。この処理については、図13の動作フローチャートを用いて後述する。なお、後述するように、ステップ718において実際にイベントデータの入替えが発生した場合は、ステップ718内において時間ソート処理終了フラグの値が0にセットされる。従って、この場合には図6のステップ602の処理に戻った時点でその判定がYESとなるため、時間ソート処理が続行される。

【0062】ステップ718の処理の後又はステップ717の判定がYESとなった後に、ステップ719で、ステップ714で判定されている現在イベントデータの種類の判定がされる。

【0063】ステップ719で現在イベントデータがト

トラック終了イベントデータ又は小節終了イベントデータであると判定された場合には、ステップ720で、フラグメント内ソート処理終了フラグの値が1にセットされる。即ち、この場合は、現在処理中のフラグメント以降に対象小節のイベントデータ群が存在しないため、ステップ720の処理の後にステップ704が実行された時点でその判定がNOとなり、フラグメント内ソート処理を終了する。

【0064】また、ステップ719で現在イベントデータがジャンプイベントデータであると判定された場合には、ステップ721で、そのジャンプイベントデータの第3ワードに格納されているジャンプ先要素番号が、サーチイベントの値としてセットされ、また、サーチ状態フラグの値が0にセットされる。即ち、この場合は、現在処理中のフラグメント以降に対象小節のイベントデータ群が存在する。このため、ステップ721の処理によってサーチイベントに新たなフラグメントの先頭のイベントデータの要素番号が設定された後に、再びステップ704と705の判定が共にYESとなった後、ステップ706以降において、配列SNG_SONGに記憶されている対象小節に含まれる新たなフラグメントに対してフラグメント内ソート処理が実行される。

【0065】以上説明した図7の動作フローチャートの処理によって、配列SNG_SONGに記憶されている対象小節内の全てのフラグメントのそれぞれに対して、図6のステップ604のフラグメント内ソート処理が1回ずつ実行される。

【0066】続いて、図8～図12は、図6のステップ605のフラグメント間ソート処理の動作フローチャートである。まず、フラグメント間ソート処理終了フラグ（CPU103内のレジスタ又はRAM105に確保される）は、フラグメント間ソート処理を、終了させるべきでない場合に値0となり、終了させるべきである場合に値1となる。このフラグの値は、ステップ801で、0に初期化される。

【0067】サーチイベント（CPU103内のレジスタ又はRAM105に確保される）は、現在処理中のイベントデータを指示するものである。このサーチイベントの値は、ステップ802で、配列SNG_SONGに記憶される対象小節内の先頭のイベントデータの要素番号にセットされる。

【0068】第1フラグメント最終イベント要素番号（CPU103内のレジスタ又はRAM105に確保される）は、フラグメント間ソート処理が実行されるべき現在処理中の2つのフラグメントのうちの第1番目のフラグメントにおける、トラック終了イベントデータ、小節終了イベントデータ、ジャンプソースイベントデータ、及びジャンプイベントデータ以外の末尾のイベントデータの要素番号を指示するものである。この値は、ステップ803で、配列SNG_SONG上の要素番号と

して出現し得ない値0xffffにセットされる。

【0069】次に、初期状態においては、ステップ804の判定がYESとなった後に、ステップ805でF1フラグ（CPU103内のレジスタ又はRAM105に確保される）の値が0にセットされる。F1フラグは、配列SNG_SONGに記憶されている対象小節に含まれる現在処理中の2つのフラグメントのうちの第1番目のフラグメントにおいて、トラック終了イベントデータ、小節終了イベントデータ、ジャンプソースイベントデータ、及びジャンプイベントデータ以外の末尾のイベントデータをサーチする処理が、終了していない場合に値0となり、終了している場合に値1となる。

【0070】続いて、図9のステップ806の判定がYESとなった後、図9のステップ807～814で、配列SNG_SONGに記憶されている対象小節に含まれる現在処理中の2つのフラグメントのうちの第1番目のフラグメントにおいて、トラック終了イベントデータ、小節終了イベントデータ、ジャンプソースイベントデータ、及びジャンプイベントデータ以外の末尾のイベントデータをサーチする処理が実行される。

【0071】まず、ステップ807で、サーチイベントにより指示される現在イベントデータが、RAM105内の配列SNG_SONGから読み出される。そして、そのイベントデータの種類が判定される。

【0072】ステップ807で現在イベントデータがトラック終了イベントデータ又は小節終了イベントデータであると判定された場合には、ステップ808で、F1フラグとフラグメント間ソート処理終了フラグの値が共に1にセットされる。即ち、この場合は、現在処理中の第1番目のフラグメント内にはソート処理を実行すべき有効な最終イベントデータは存在せず、かつ第1番目のフラグメント以降に対象小節のイベントデータ群が存在しないため、ステップ808の処理の後にステップ806が実行された時点でその判定がNOとなり、後述する図10のステップ815が実行された後、ステップ816に対応する後述する図11の動作フローチャートが実行される時点でステップ1101の判定がNOとなり、更に、図10のステップ817の判定がNO、図8のステップ804の判定がNOとなって、フラグメント間ソート処理を終了する。

【0073】また、図9のステップ807で現在イベントデータがジャンプイベントデータであると判定された場合には、ステップ809で、第1フラグメント最終イベント要素番号の値がステップ803で初期設定されたままの値0xffffであるか否かが判定される。

【0074】今、配列SNG_SONG内の現在処理中の2つのフラグメントのうちの第1番目のフラグメント内に、トラック終了イベントデータ、小節終了イベントデータ、ジャンプソースイベントデータ、及びジャンプイベントデータ以外のイベントデータが1つも存在して

いなければ、第1フラグメント最終イベント要素番号の値は初期設定値0x f f f fのままとなり、ステップ809の判定がYESとなる。この場合には、ステップ811で、そのジャンプイベントデータの第3ワードに格納されている新たなフラグメントを示すジャンプ先要素番号が、サーチイベントの値としてセットされる。その後、再びステップ806の判定がYESとなることにより、ステップ807～814において、配列SNG_SONGに記憶されている対象小節に含まれる新たなフラグメントが現在処理中の2つのフラグメントのうちの第1番目のフラグメントとされて、処理が続行される。

【0075】更に、ステップ807で現在イベントデータがトラック終了イベントデータ、小節終了イベントデータ、及びジャンプイベントデータ以外のイベントデータであると判定された場合において、更にステップ812で現在イベントデータがジャンプソースイベントデータであると判定された場合には、ステップ814で、単純にサーチイベントが指示する配列SNG_SONGの要素番号がインクリメントされた後に、再びステップ806からステップ807の処理に戻る。

【0076】また、ステップ807で現在イベントデータがトラック終了イベントデータ、小節終了イベントデータ、及びジャンプイベントデータ以外のイベントデータであると判定された場合において、更にステップ812で現在イベントデータがジャンプソースイベントデータ以外の通常のイベントデータであると判定された場合には、ステップ813で、サーチイベントの値が第1フラグメント最終イベント要素番号としてセットされる。そして、ステップ814で、サーチイベントが指示する配列SNG_SONGの要素番号がインクリメントされた後に、再びステップ806からステップ807の処理に戻る。

【0077】以上のようにして第1フラグメント最終イベント要素番号がインクリメントされてゆき、最後にステップ807でジャンプイベントデータが検出された時点において第1フラグメント最終イベント要素番号にセットされている要素番号が、配列SNG_SONGに記憶されている対象小節に含まれる現在処理中の2つのフラグメントのうちの第1番目のフラグメントにおいて見つかった、トラック終了イベントデータ、小節終了イベントデータ、ジャンプソースイベントデータ、及びジャンプイベントデータ以外の末尾のイベントデータの要素番号となる。その後、ステップ809の判定がNOとなつて、第1番目のフラグメントにおけるサーチ処理を終了させるべく、ステップ810で、F1フラグの値が1にセットされる。そして、ステップ811で、そのジャンプイベントデータの第3ワードに格納されている新たなフラグメントを示すジャンプ先要素番号が、サーチイベントの値としてセットされる。

【0078】この結果、ステップ811に続いて実行さ

れるステップ806の判定がNOとなり、図10のステップ815でF2フラグ（CPU103内のレジスタ又はRAM105に確保される）の値が0にセットされる。F2フラグは、配列SNG_SONGに記憶されている対象小節に含まれる現在処理中の2つのフラグメントのうちの第2番目のフラグメントにおいて、トラック終了イベントデータ、小節終了イベントデータ、ジャンプソースイベントデータ、及びジャンプイベントデータ以外の先頭のイベントデータをサーチする処理が、終了していない場合に値0となり、終了している場合に値1となる。

【0079】続いて、図10のステップ816で、配列SNG_SONGに記憶されている対象小節に含まれる現在処理中の2つのフラグメントのうちの第2番目のフラグメントにおいて、トラック終了イベントデータ、小節終了イベントデータ、ジャンプソースイベントデータ、及びジャンプイベントデータ以外の先頭のイベントデータをサーチする処理が実行される。

【0080】図11は、図10のステップ816の処理の詳細を示す動作フローチャートである。まず、最初はステップ1101の判定がYESとなる。

【0081】次に、ステップ1102で、サーチイベントにより指示される現在イベントデータが、RAM105内の配列SNG_SONGから読み出される。そして、そのイベントデータの種類の判定される。

【0082】ステップ1102で現在イベントデータがトラック終了イベントデータ又は小節終了イベントデータであると判定された場合には、ステップ1103で、フラグメント間ソート処理終了フラグの値が1にセットされる。即ち、この場合は、現在処理中の2つのフラグメントのうちの第2番目のフラグメント内にはソート処理を実行すべき有効な先頭イベントデータは存在せず、かつ第2番目のフラグメント以降に対象小節のイベントデータ群が存在しないため、ステップ1103の処理の後にステップ1101が実行された時点でその判定がNOとなり、図11の動作フローチャートによって示される図10のステップ816の処理を終了する。

【0083】また、ステップ1102で現在イベントデータがジャンプイベントデータであると判定された場合には、ステップ1104で、そのジャンプイベントデータの第3ワードに格納されているジャンプ先要素番号が、サーチイベントの値としてセットされる。即ち、この場合には、現在処理中の2つのフラグメントのうちの第2番目のフラグメント内にはソート処理を実行すべき有効なイベントデータは存在せず、かつその第2番目のフラグメント以降に対象小節のイベントデータ群が存在する。このため、ステップ1104の処理によってサーチイベントに新たなフラグメントの先頭のイベントデータの要素番号が設定される。その後、再びステップ1101の判定がYESとなることにより、ステップ110

2~1108において、配列SNG_SONGに記憶されている対象小節に含まれる新たなフラグメントが現在処理中の2つのフラグメントのうちの第2番目のフラグメントとされて、処理が続行される。

【0084】更に、ステップ1102において現在イベントデータがトラック終了イベントデータ、小節終了イベントデータ、及びジャンプイベントデータ以外のイベントデータであると判定された場合において、更にステップ1105で現在イベントデータがジャンプソースイベントデータであると判定された場合には、ステップ1108で、単純にサーチイベントが指示する配列SNG_SONGの要素番号がインクリメントされた後に、再びステップ1101からステップ1102の処理に戻る。

【0085】また、ステップ1102で現在イベントデータがトラック終了イベントデータ、小節終了イベントデータ、及びジャンプイベントデータ以外のイベントデータであると判定された場合において、更にステップ1105で現在イベントデータがジャンプソースイベントデータ以外の通常のイベントデータであると判定された場合には、ステップ1106で、サーチイベントの値が第2フラグメント先頭イベント要素番号としてセットされる。この値が、配列SNG_SONGに記憶されている対象小節に含まれる現在処理中の2つのフラグメントのうちの第2番目のフラグメントにおいて最初に見つかった、トラック終了イベントデータ、小節終了イベントデータ、ジャンプソースイベントデータ、及びジャンプイベントデータ以外の先頭のイベントデータの要素番号となる。続いて、第2番目のフラグメントにおけるサーチ処理を終了させるべく、ステップ1107で、F2フラグの値が1にセットされた後、ステップ1108で、サーチイベントが指示する配列SNG_SONGの要素番号がインクリメントされる。そして、ステップ1106~1108が実行された後にステップ1101が再び実行される時点で、その判定がNOとなり、図11によって示される図10のステップ816の処理を終了する。

【0086】以上のようにして、配列SNG_SONGに記憶されている対象小節に含まれる現在処理中の2つのフラグメントにつき、図9のステップ807~814で、第1番目のフラグメントの末尾のイベントデータをサーチする処理が成功し、かつ、図11によって示される図10のステップ816で、第2番目のフラグメントの先頭のイベントデータをサーチする処理に成功すると、ステップ817の判定がYESとなった後、ステップ818で、第1番目のフラグメントの末尾のイベントデータと第2番目のフラグメントの先頭のイベントデータについて、イベント移動処理が実行される。

【0087】図12は、図10のステップ818のイベント移動処理の動作フローチャートである。まず、ステ

ップ1201で、RAM105内の配列SNG_SONGから、第1フラグメント最終イベント要素番号によって指示される要素番号のイベントデータが読み出されると共に、第2フラグメント先頭イベント要素番号によって指示される要素番号のイベントデータが読み出される。そして、第1フラグメント最終イベント要素番号に対応するイベントデータの第2ワードの値が、第2フラグメント先頭イベント要素番号に対応するイベントデータの第2ワードの値より大きいかな否か、即ち、第1フラグメント最終イベント要素番号に対応するイベントデータが第2フラグメント先頭イベント要素番号に対応するイベントデータよりも後ろに配置されるべきかな否かが判定される。

【0088】ステップ1201の判定がYESの場合には、ステップ1202で、第1フラグメント最終イベント要素番号に対応するイベントデータと第2フラグメント先頭イベント要素番号に対応するイベントデータの記憶位置が入れ替えられる。その後、ステップ1203及び1204で、フラグメント間ソート処理終了フラグ及び時間ソート処理終了フラグの値が、共に0にセットされる。このように、実際にイベントデータの入替えが発生した場合には、時間ソート処理終了フラグの値が0にセットされるため、前述した図6のステップ604のフラグメント内ソート処理とステップ605のフラグメント間ソート処理が終了した後にステップ602の判定に戻った時点で、その判定が再びYESとなり、時間ソート処理が続行される。

【0089】一方、ステップ1201の判定がNOの場合には、第1フラグメント最終イベント要素番号に対応するイベントデータと第2フラグメント先頭イベント要素番号に対応するイベントデータの入替えは実行されず、ステップ1205で、フラグメント間ソート処理終了フラグの値が0にセットされる。

【0090】以上のようにして、配列SNG_SONGに記憶される対象小節に含まれる現在処理中の2つのフラグメントについて、フラグメント間ソート処理が実現される。

【0091】ステップ1204又は1205の処理の後、再び図8のステップ804の処理に戻り、その判定がYESとなることにより、ステップ805以降で、配列SNG_SONGに記憶されている対象小節に含まれる新たな2つのフラグメントにつき、第1番目のフラグメントの末尾のイベントデータをサーチする処理と、第2番目のフラグメントの先頭のイベントデータをサーチする処理と、サーチされた2つのイベントデータに対するイベント移動処理の実行が試みられる。この場合、サーチイベントの値は今まで第2番目であったフラグメントの先頭のイベントデータを指示しているため、今まで第2番目として処理されたフラグメントが新たな第1番目のフラグメントとして処理される。そして、それに続

く対象小節内のフラグメントが第2番目のフラグメントとして処理される。

【0092】以上の処理が繰返される結果、対象小節の末尾のフラグメントが第1番目のフラグメントとして処理される過程において、ステップ807でトラック終了イベントデータ又は小節終了イベントデータが検出される。これにより、ステップ808で、F1フラグとフラグメント間ソート処理終了フラグの値が共に1にセットされる。即ち、この場合は、現在処理中の第1番目のフラグメント内にはソート処理を実行すべき有効な最終イベントデータは存在せず、かつ第1番目のフラグメント以降に対象小節のイベントデータ群が存在しないため、ステップ808の処理の後にステップ806が実行された時点でその判定がNOとなり、後述する図10のステップ815が実行された後、ステップ816に対応する後述する図11の動作フローチャートが実行される時点でステップ1101の判定がNOとなり、更に、図10のステップ817の判定がNO、図8のステップ804の判定がNOとなって、フラグメント間ソート処理を終了する。

【0093】最後に、図13は、図7のステップ718において実行される、配列SNG_SONGに記憶されている対象小節に含まれる現在処理中のフラグメント内の、フラグメント先頭イベントとフラグメント終了イベントにより指示される範囲のイベントデータ群に対する時間ソート処理を示す動作フローチャートである。

【0094】ここでは、フラグメント先頭イベントとフラグメント終了イベントにより指示される範囲のイベントデータ群が、各イベントデータの第2ワードのクロック値に関してソートされる。

【0095】以下の説明では、図13の動作フローチャートについて説明する前に、ソートアルゴリズムの代表例と、本実施例で採用する改良アルゴリズムの原理について簡単に説明する。なお、以下で説明するバブルソートアルゴリズムとコムソートアルゴリズムの詳細な原理については、文献「バブル・ソートが簡単な手直しで劇的に速くなる」(スティーブン・レイシー、リチャード・ボックス)(日経BP社発行、日経バイト/1991年11月号、p. 305~p. 312)に記載されている。

【0096】ソートアルゴリズムの最も代表的なものとして、バブルソートアルゴリズムが知られている。このアルゴリズムにおいては、例えば昇べきの順にソートが行われる場合には、まず、第1番目のイベントデータ値(第2バイトの値)と第2番目のイベントデータ値の大小関係が比較され、第1番目のイベントデータ値の方が大きければ両者のイベントデータが入れ替えられる。次に、新たな第2番目のイベントデータ値と第3番目のイベントデータ値の大小関係が同様に比較され入替えが実行される。このようにして、最終イベントデータまで同

様の比較・入替え処理が実行される。この最終データまでの1回の処理単位を、ストロークと呼ぶ。続いて、再び、第1番目のイベントデータから最終イベントデータに向かって第2ストローク目の処理が繰返される。そして、このストローク処理が、イベントデータの入替えが発生しなくなるまで繰返される。

【0097】以上のバブルソートアルゴリズムは、隣り同士のイベントデータ値の比較・入替え処理の繰返しによって実現されるため、処理プログラムが簡単に実現できる反面、イベントデータは1回の比較・入替え処理によって1アドレス分だけしか移動できないため、例えばフラグメントの後部に位置するイベントデータがフラグメントの先頭付近に移動させられる場合には処理時間が長くなってしまう。

【0098】そこで、そのようなバブルソートアルゴリズムの欠点を改良するアルゴリズムとして、コムソートアルゴリズムが知られている。このアルゴリズムでは、バブルソートアルゴリズムのように隣接したイベントデータ値同士について比較・入替えが実行されるのではなく、1以上離れた間隔(以下、この間隔をギャップという)のイベントデータ値同士について、比較・入替えが実行される。

【0099】具体的には、次のようなコムソートアルゴリズムが最適であることが実験的に確かめられている。即ち、まず、最初のストローク処理(第1番目のイベントデータから最終イベントデータまでの1回の処理)におけるギャップの値は、ソート処理されるイベントデータの要素数を1.3で割って得られる値とされる。以下、ストロークが進む毎に、前回のギャップの値を1.3で割って得られる値が新たなギャップの値とされる。この過程で、ギャップの値が9又は10となった時点で、ギャップの値が強制的に11に置き換えられて処理が続行される。商が1未満になると、それ以降のギャップの値は1とされる。そして、ギャップの値が1となり、かつストローク処理においてイベントデータの入替えが発生しなくなった時点で、ソート処理が終了する。

【0100】以上のコムソートアルゴリズムが、最も高速なソートアルゴリズムとして知られている。しかし、上述のコムソートアルゴリズムでは、ストロークが進む毎にソート処理されるべき要素数又は前回のギャップ値を1.3で割ることにより新たなギャップの値を算出する処理が必要となる。このことは、上記除算処理のために小数演算用コプロセッサが必要になるということを意味するため、ハードウェア規模が大きくなってしまいうという欠点がある。

【0101】そこで、本実施例では、コムソートアルゴリズムを基本とし、値1.3による除算処理を、値3の乗算処理と値4の除算処理で置き換えたアルゴリズムが採用される。即ち、値1.3による除算処理が値(3/4)による除算処理に置き換えられたことになる。これ

により、小数演算用コプロセッサが不要となり、ハードウェア規模の増大を抑えることができる。なお、このようにギャップ値が代用されても、ソート時間の増大はそれほど大きくならない。

【0102】図13は、上述のアルゴリズムに基づく図7のステージ718の処理の詳細な動作フローチャートである。図13の動作フローチャートにおいて、まず、SIZE (CPU103内のレジスタ又はRAM105に確保される) には、ソート処理の対象となるフラグメントの要素数がセットされる。この数は、(フラグメント終了イベントーフラグメント先頭イベント+1) として算出できる。

【0103】次に、GAP (CPU103内のレジスタ又はRAM105に確保される) には、前述のギャップの値がセットされる。また、TOP (CPU103内のレジスタ又はRAM105に確保される) には、後述するイベント移動ループ処理時の上限のイベントの要素値がセットされる。

【0104】更に、COUNTER (CPU103内のレジスタ又はRAM105に確保される) には、後述する移動イベントカウンタ値がセットされる。図13の動作フローチャートにおいて、まず、ステップ1301で、GAPにSIZEの値がセットされる。次に、前述した除算処理の簡略化原理に基づき、ステップ1302において、GAPの値に整数値3を乗算する整数乗算処理が実行され、更に、その乗算結果を整数値4で除算する整数除算処理が実行される。この除算処理は、実際には、2ビット右シフト演算として実現できる。この演算結果が新たなGAPの値としてセットされる。

【0105】次に、ステップ1303で新たに算出されたGAPの値が判定され、GAPの値が9又は10の場合には、前述したコムソート11のアルゴリズムの原理に基づいて、ステップ1304で、GAPの値が11に修正される。また、GAPの値が0となってしまった場合には、やはり前述したコムソート11のアルゴリズムの原理に基づいて、ステップ1305で、GAPの値が1に修正される。GAPの値が9、10、又は0の何れでもない場合には、GAPの値の修正は行われない。

【0106】次に、ステップ1306で、COUNTERの値が0にセットされる。続いて、ステップ1307で、第1イベント要素番号 (CPU103内のレジスタ又はRAM105に確保される) に、図7の動作フローチャートにおいて求まっているフラグメント先頭イベントにセットされている要素番号がセットされる。この第1イベント要素番号は、後述するイベント移動ループ処理において比較される2つのイベントデータのうち要素番号が小さい方のイベントデータに対応する。

【0107】次に、TOPの値が、次の数1式により算出されセットされる。

【0108】

【数1】

TOP=フラグメント先頭イベント+SIZE-GAP
このTOPの値は、次に説明するイベント移動ループ処理においてインクリメントすることのできる第1イベント要素番号の上限値に対応する。

【0109】続いて、ステップ1309~1314→1315→1309のイベント移動ループ処理が実行される。ここでは、1ストローク分のイベントデータの比較・入替え処理が実行される。

【0110】まず、ステップ1309では、第2イベント要素番号 (CPU103内のレジスタ又はRAM105に確保される) の値が、次の数2式によって算出されセットされる。

【0111】

【数2】

第2イベント要素番号=第1イベント要素番号+GAP
この第2イベント要素番号は、イベント移動ループ処理において比較される2つのイベントデータのうち要素番号が大きい方のイベントデータに対応する。

【0112】ステップ1310では、RAM105の配列SNG_SONG (図3(c)) に記憶される第1イベント要素番号に対応するイベントデータの第2ワードの小節先頭からのクロック値が、同じく第2イベント要素番号に対応するイベントデータの第2ワードの小節先頭からのクロック値よりも大きいのか否か、即ち、第1イベント要素番号に対応するイベントデータが第2イベント要素番号に対応するイベントデータよりも後ろに配置されるべきか否かが判定される。

【0113】ステップ1310の判定がYESの場合には、ステップ1311で、第1イベント要素番号に対応するイベントデータと第2イベント要素番号に対応するイベントデータの記憶位置が入れ替えられる。その後、ステップ1312で、COUNTERの値がインクリメントされる。更に、ステップ1313で、時間ソート処理終了フラグの値が、0にセットされる。このように、実際にイベントデータの入替えが発生した場合には、時間ソート処理終了フラグの値が0にセットされるため、前述した図6のステップ604のフラグメント内ソート処理とステップ605のフラグメント間ソート処理が終了した後にステップ602の判定に戻った時点で、その判定が再びYESとなり、時間ソート処理が続行される。

【0114】一方、ステップ1310の判定がNOの場合には、ステップ1311~1313の処理は実行されない。続いて、ステップ1314で、第1イベント要素番号の値がTOPの値より小さいか否かが判定される。

【0115】ステップ1314の判定がYESなら、ステップ1315で第1イベント要素番号の値がインクリメントされた後、ステップ1309で第1イベント要素番号からGAPの値分だけ離れた新たな第2イベント要

奏番号の値が算出され、ステップ1310～1313で、両者に対する比較・入替えの処理が実行される。

【0116】以上のステップ1309～1315の処理がステップ1314の判定がNOとなるまで繰り返されることにより、1ストローク分のイベントデータの比較・入替え処理が実行される。

【0117】次に、ステップ1314の判定がNOとなり1ストローク分のイベントデータの比較・入替え処理が終了すると、ステップ1315で、COUNTERの値が0でないか（即ち、直前にイベントデータの入替えが行われたか）、又はGAPの値が1より大きいかが判定される。

【0118】ステップ1316の判定がYESなら、ステップ1302～1305に戻って新たなストロークのためのGAPの値が算出され、ステップ1306～1308の処理の後、前述したステップ1309～1315のイベント移動ループ処理によって、新たなストロークに対するイベントデータの比較・入替え処理が繰り返される。

【0119】GAPの値が1となり、かつ前述したイベント移動ループ処理によってイベントデータの入替えが発生しなくなり、ステップ1316の判定がNOとなった時点で、図13によって示される図7のステップ718のフラグメント内のソート処理が終了する。

【0120】

【発明の効果】本発明によれば、例えば楽曲データの再生処理において、非常に高速な1又は複数小節単位の早送り再生又は巻き戻し再生を簡単な制御で実現することが可能となる。

【0121】また、楽曲データの編集処理が実行される場合に、例えば小節単位の移動、既存の小節への新たな楽曲データの追加、その他、楽曲データの挿入、削除等を、簡単なデータ操作で実現することが可能となる。

【0122】このように、楽曲データが、第1の記憶手段、第2の記憶手段、又は第3の記憶手段を介して階層的に管理されることにより、楽曲データの再生処理、編

集処理等の制御を簡単に実行することが可能となる。

【図面の簡単な説明】

【図1】本発明の実施例の構成図である。

【図2】RAM105に記憶される楽曲データのデータフォーマットを示す図（その1）である。

【図3】RAM105に記憶される楽曲データのデータフォーマットを示す図（その2）である。

【図4】ノートイベントデータとコントロールイベントデータのデータフォーマットを示す図である。

【図5】ジャンプイベントデータ群のデータフォーマットを示す図である。

【図6】時間ソート処理の全体動作フローチャートである。

【図7】フラグメント内ソート処理の動作フローチャートである。

【図8】フラグメント間ソート処理の動作フローチャート（その1）である。

【図9】フラグメント間ソート処理の動作フローチャート（その2）である。

【図10】フラグメント間ソート処理の動作フローチャート（その3）である。

【図11】第2フラグメント先頭イベントサーチ処理の動作フローチャートである。

【図12】イベント移動処理の動作フローチャートである。

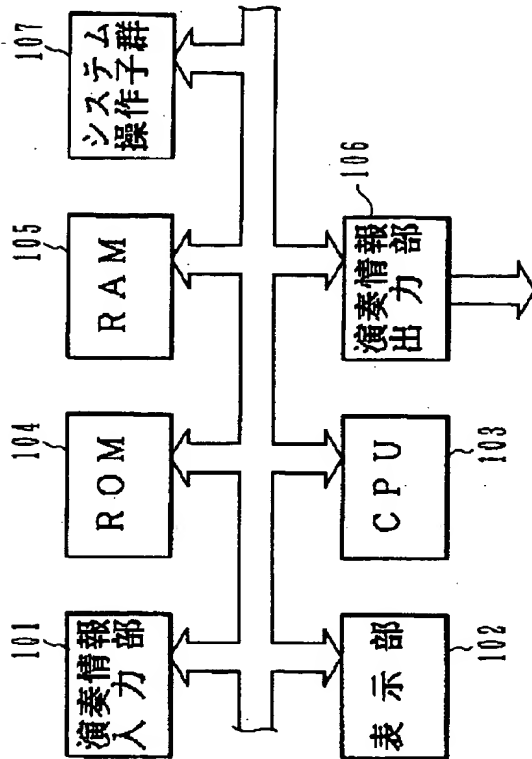
【図13】フラグメント内のイベントソート処理を示す動作フローチャートである。

【符号の説明】

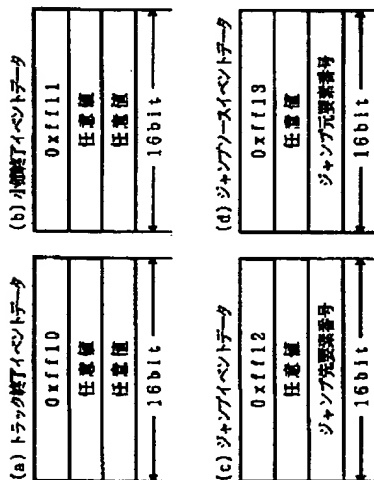
101	演奏情報入力部
102	表示部
103	CPU
104	ROM
105	RAM
106	演奏情報出力部
107	システム操作子群

【図1】

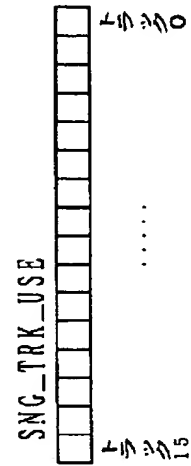
本発明の実施例の構成図



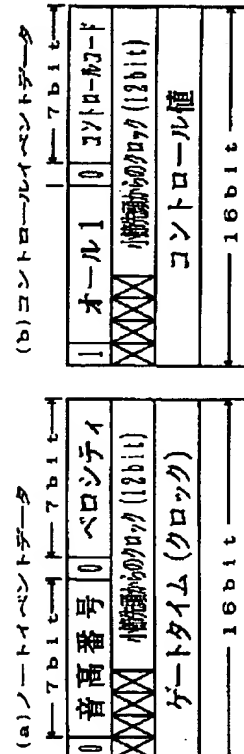
【図5】

ジャンプイベントデータ群の
データフォーマットを示す図

【図2】

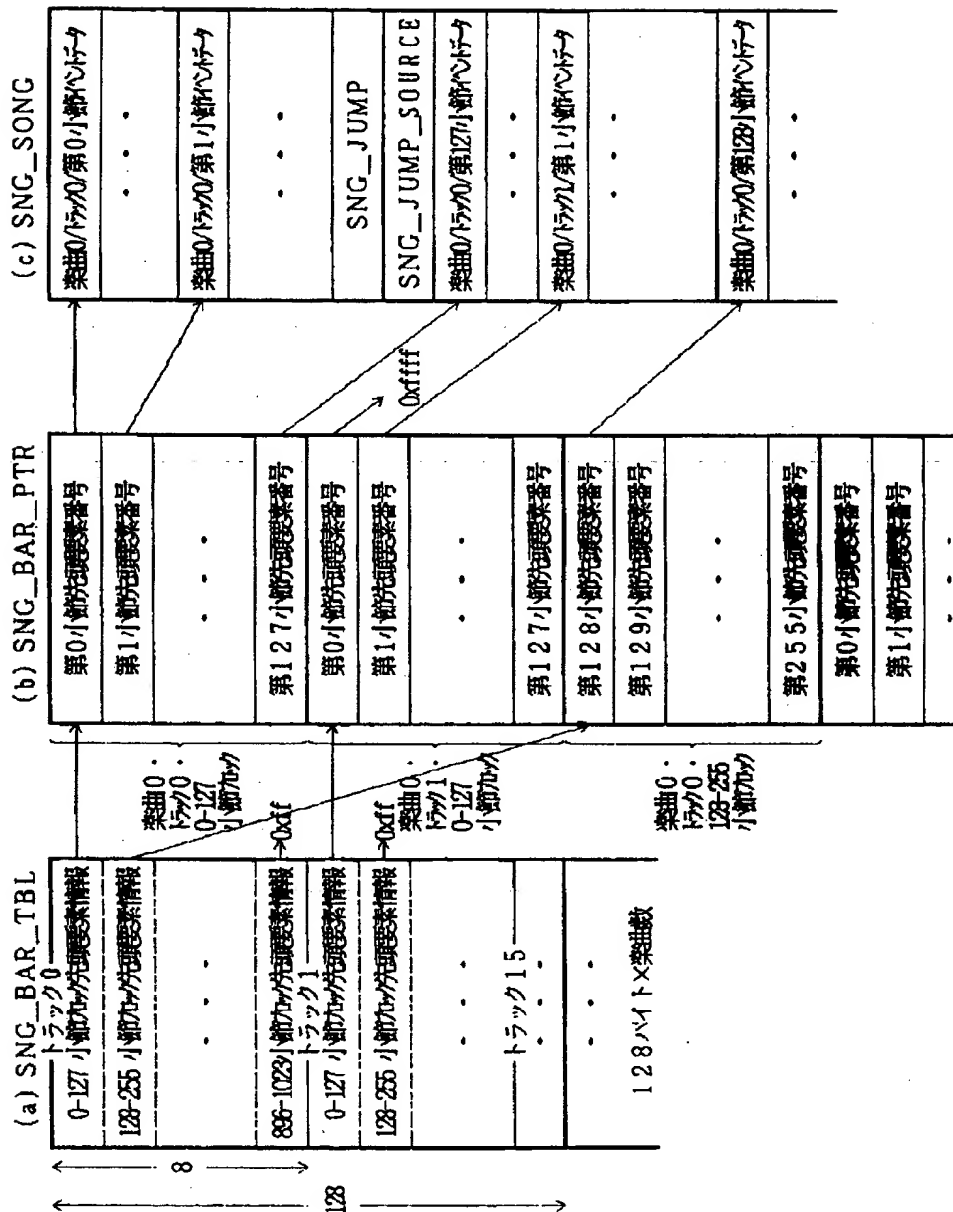
RAM105に記憶される楽曲データの
データフォーマットを示す図(その1)

【図4】

ノートイベントデータとコントロールイベントデータの
データフォーマットを示す図

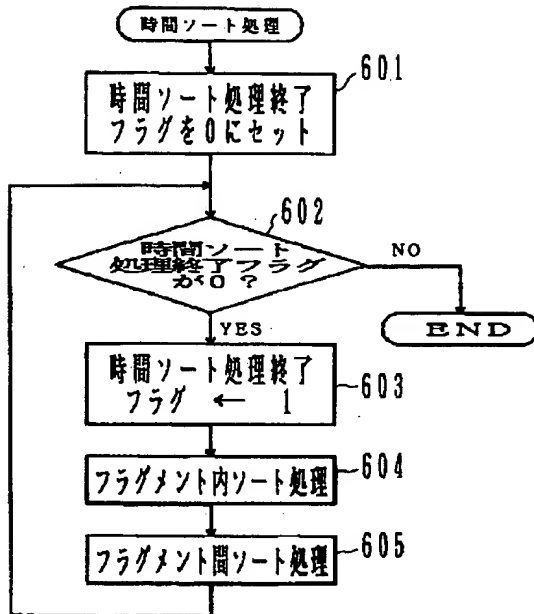
【図3】

RAM105に記憶される楽曲データの
データフォーマットを示す図(その2)



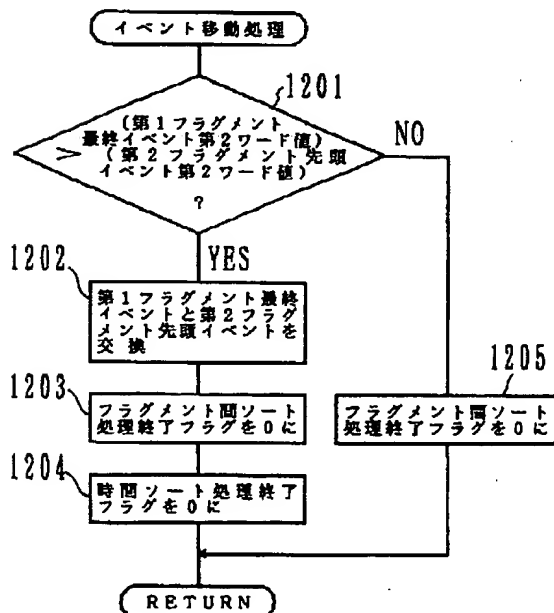
【図6】

時間ソート処理の全体動作フローチャート



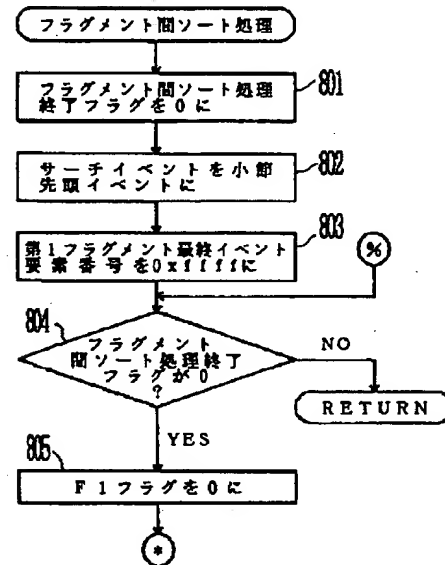
【図12】

イベント移動処理の動作フローチャート



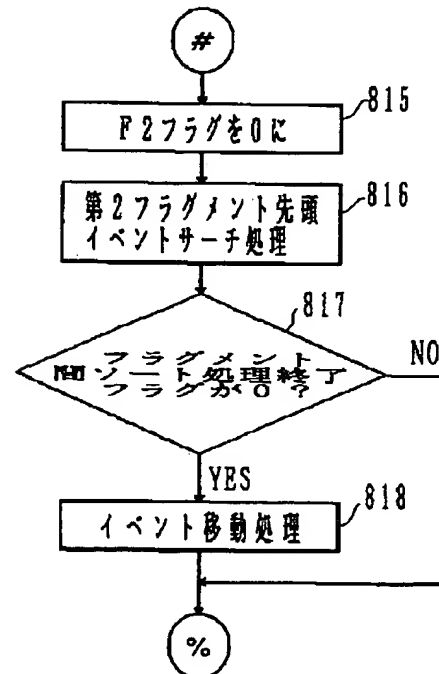
【図8】

フラグメント間ソート処理の動作フローチャート(その1)



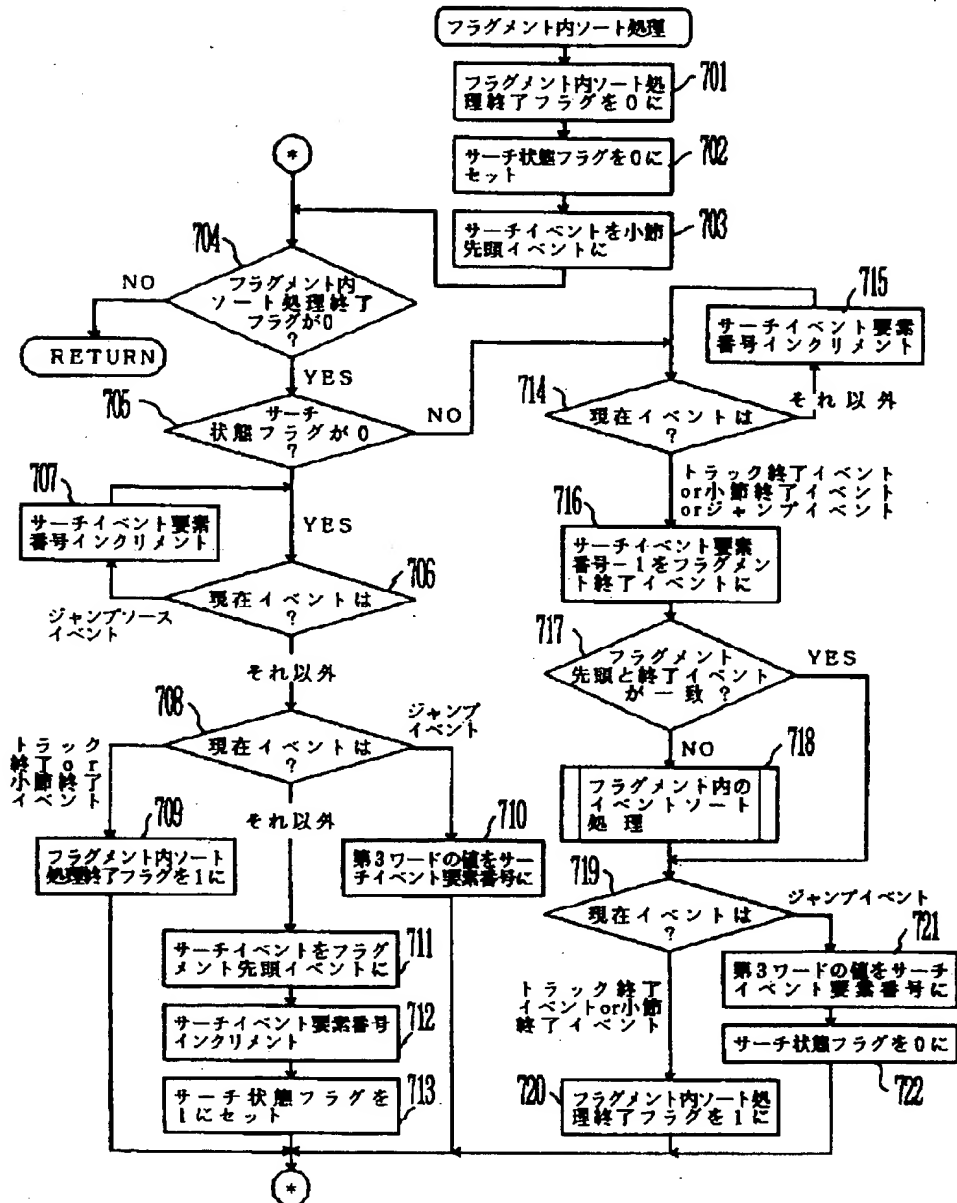
【図10】

フラグメント間ソート処理の動作フローチャート(その3)



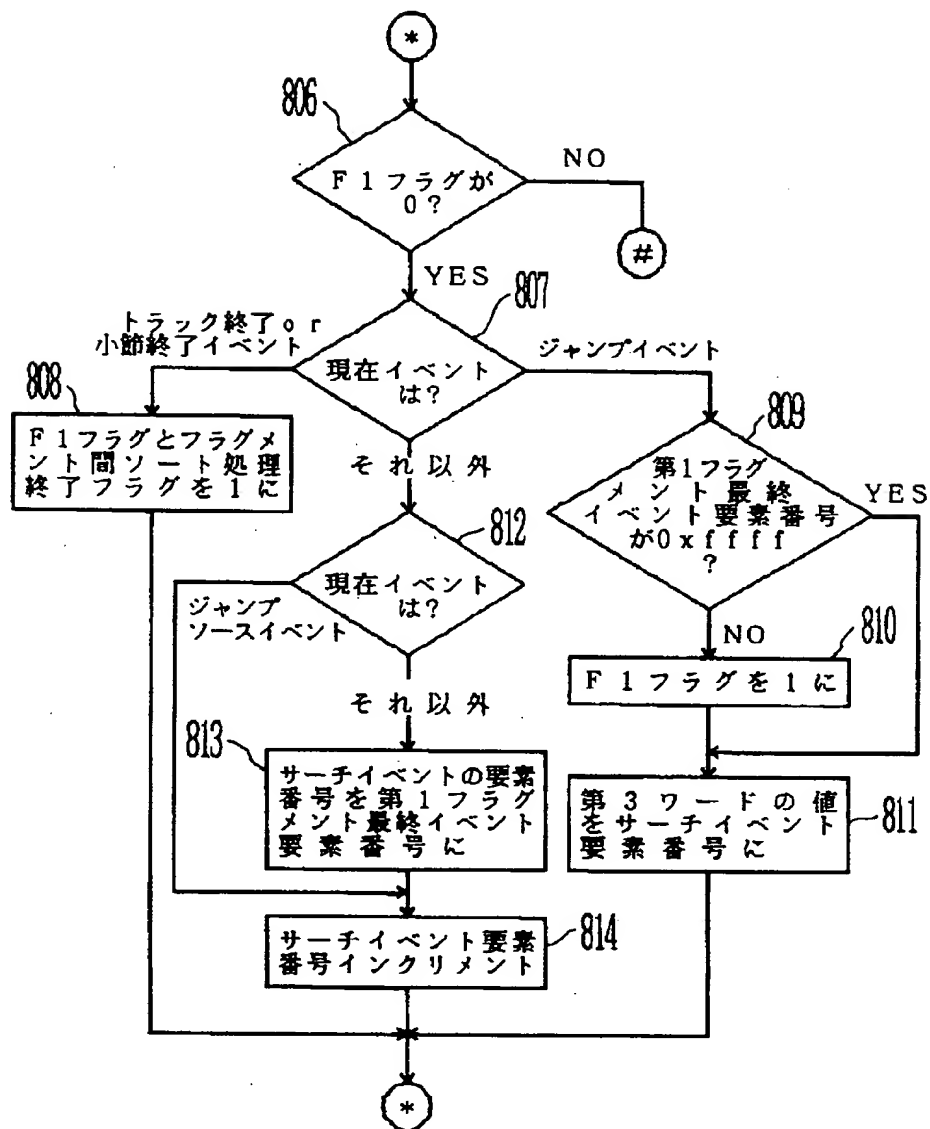
【図7】

フラグメント内ソート処理の動作フローチャート



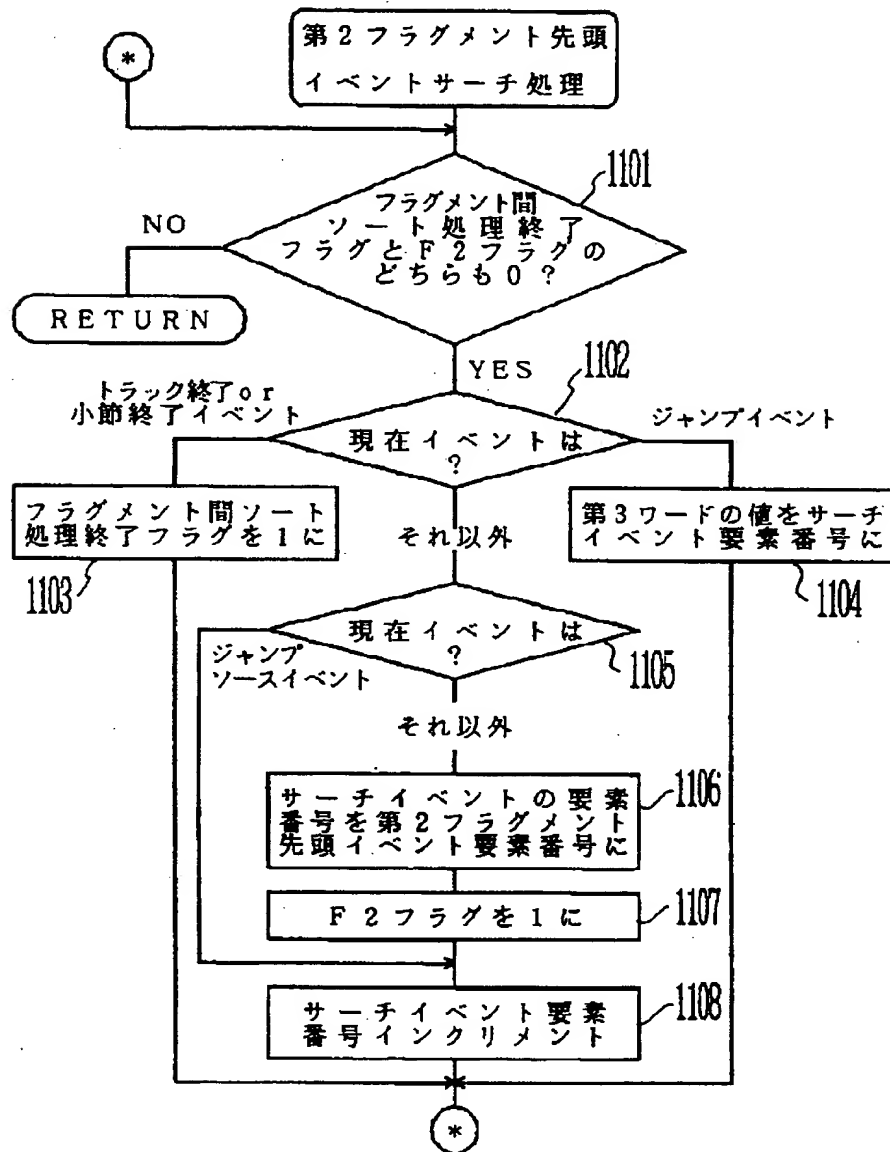
【図9】

フラグメント間ソート処理の動作フローチャート (その2)



【図11】

第2フラグメント先頭イベントサーチ処理の動作フローチャート



【図13】

フラグメント内のイベントソート処理を示す動作フローチャート

